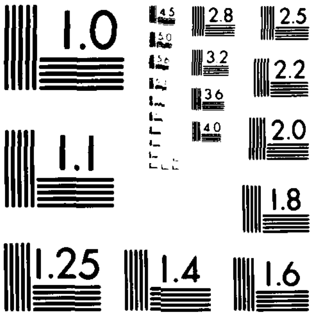
AD-A099 726 DEFENSE MAPPING AGENCY WASHINGTON DC F/G 9/2
SYNCHRONOUS NETS FOR SINGLE INSTRUCTION STREAM - MULTIPLE DATA --ETC(U)
MAY 81 A J KRYGIEL

UNCLASSIFIED NL

1 of 1
AD-A
099726

END
DATE
FILMED
6-81
DTIC



MICROCOPY RESOLUTION TEST CHART

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

①

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER N/A	2. GOVT ACCESSION NO. AD-A099726	3. RECIPIENT'S CATALOG NUMBER N/A
4. TITLE (and Subtitle) Synchronous Nets for Single Instruction Stream - Multiple Data Stream Computers.		5. TYPE OF REPORT & PERIOD COVERED N/A 6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) (10) Dr./Annette J./Krygiel		8. CONTRACT OR GRANT NUMBER(s) N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS Defense Mapping Agency (STT) Bldg. 56, US Naval Observatory, Wash, DC 20305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS N/A
11. CONTROLLING OFFICE NAME AND ADDRESS Same as above (12) 35		12. REPORT DATE 13. NUMBER OF PAGES 20 + 13 Figure
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) N/A (11 28 May 81		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

(6

16. DISTRIBUTION STATEMENT (of this Report)

Dissemination to the public and also to the 1981 International Conference on Parallel Processing

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

N/A

18. SUPPLEMENTARY NOTES

To be presented at 1981 International Conference on Parallel Processing

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SIMD Petri Nets	Sychronous Nets	S Nets
Parallel Processors	Models of Parallel Compertation	
Array Processors	Graph Based Modeling	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Synchronous Nets, or S Nets, are developed as a modeling tool particularized for describing processes on Single Instruction Stream - Multiple Data Stream (SIMD) computers. S Nets are a modification of Petri Nets, using transitions and places to model events and conditions. However S Nets introduce vector-mask places to model the conditions of the array resources of SIMD machines. These places are distinguished from scalar places which model the scalar resources. S Nets also introduce a new kind of transition. One type correlates with the Petri Net transition, but the mask firing transition is particularized to the SIMD

UNCLASSIFIED 390420

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT Continued- environment, modeling the inherent capability of a computation executing on a SIMD machine to alter the participation of the vector aggregates in successor events.

ABSTRACT

SYNCHRONOUS NETS FOR SINGLE INSTRUCTION STREAM - MULTIPLE DATA STREAM COMPUTERS

by Annette J. Krygiel

Synchronous Nets, or S Nets, are developed as a modeling tool particularized for describing processes on Single Instruction Stream - Multiple Data Stream (SIMD) computers. S Nets are a modification of Petri Nets, using transitions and places to model events and conditions. However S Nets introduce vector-mask places to model the conditions of the array resources of SIMD machines. These places are distinguished from scalar places which model the scalar resources. S Nets also introduce a new kind of transition. One type correlates with the Petri Net transition, but the mask firing transition is particularized to the SIMD environment, modeling the inherent capability of a computation executing on a SIMD machine to alter the participation of the vector aggregates in successor events.

SYNCHRONOUS NETS FOR SINGLE INSTRUCTION STREAM - MULTIPLE DATA STREAM COMPUTERS

1. INTRODUCTION

This paper is concerned with the problem of mapping algorithms onto certain classes of parallel processors to exploit parallelism in the algorithm to the maximum extent supportable by the machine on which it is implemented. The approach taken is one of providing a tool -- a graph-based modeling system called Synchronous Nets or S Nets -- to describe such an implementation. This not only supports examination of the possibilities for parallelism in the algorithm and the machine, but also provides a means of describing and documenting results.

1.1 SIMD Architectures

The processors of interest are of the single instruction stream-multiple data stream (SIMD) architectures as described by Flynn [1] who distinguishes four classes:

Single instruction stream - single data stream (SISD)

Single instruction stream - multiple data stream (SIMD)

Multiple instruction stream - single data stream (MISD)

Multiple instruction stream - multiple data stream (MIMD)

Figure 1 illustrates an SIMD architecture, which typically consists of a control unit with its own memory, and (possibly) some limited processing capability; an array or vector unit consisting of N Processing Elements (PEs) and at least N memories (PEMs); and an interconnection network for interprocessor communication. Associated with each PE is some indicator (mask) for signaling participation or non-participation in instructions. In implementation, usually a conventional sequential machine is attached to the control units, i.e. a mini-host. In some instances then, the SIMD "system" can admit a multiplicity in the instruction stream, where one of the instruction streams is provided either by the control "processor" or the mini-host, in addition to that utilized by the array resource.

A Multiple-SIMD (MSIMD) architecture is configured as two or more independent SIMD machines, each with its own control unit, array unit, etc., and with one interconnection network. These SIMD components have the ability to perform synchronously or asynchronously, and using the same instruction stream or different instruction streams. Such an architecture is illustrated by Figure 2.

1.2 Potential of SIMD Architectures

SIMD machines are considered "special purpose". They perform spectacularly on problems to which they are well suited [2]. For instance, an IBM 360/75 system augmented with a STARAN showed a 57:1 gain for adaptive clustering algorithms which accomplish pattern recognition on LANDSAT imagery using statistical measurements [3]. Krygiel reported a 70:1 improvement with STARAN over a UNIVAC 1108 for a 1024 point Fast Hadamard Transform [4]. Daley's implementation of a weather prediction model on ILLIAC IV required 11 minutes in comparison to 27 hours on an IBM 360/67 [5].

To derive high performance, the application should have a high degree of parallelism, with the topology of the algorithm consistent with the topology of the machine. Unfortunately there are no simple means to gauge this desired isomorphism. Modeling is one approach that can be employed. S Nets were specifically developed to accomplish this, and are a modification of Petri Nets [6, 7, 8] supplying constructs particularized to SIMD (and MSIMD architectures).

2. DEFINITION OF SYNCHRONOUS NETS

2.1 SYSTEM OVERVIEW

A Synchronous Net, or S Net, is a directed graph with a marking and a set of descriptors [9]. The vertices of the graph are vector-mask places, scalar places, and transitions. Scalar places and vector-mask places are connected with arcs to transitions and vice versa. A marking associates a non-negative integer with each scalar place, and associates a tuple of non-negative integers with each vector-mask place. The non-negative number is called the number of tokens. Descriptors are associated with each transition and characterize the behavior of the transition.

The S Net exhibits dynamic behavior resulting from the firing of transitions. Only enabled transitions may fire, and a transition is enabled (also called firable) if the marking of its immediate antecedent places (input places) meets certain requirements. The firing of a transition changes the marking on its input places and its immediate successor places (output places) in accordance with the transition's descriptor.

As with Petri Nets S Nets use transitions to model events and places to model conditions with arcs representing the paths allowed for passage of control. The firing of a transition models the occurrence of an event; tokens in a place can model the holding of a condition; however there are particular requirements associated with holdings for vector-mask places other than just the presence of tokens.

Intrinsic and original to this system is the notion of vector-mask places, which can be used to model aggregates of logically associated and homogeneous conditions whose initial and ceasing events are synchronized, i.e., the conditions of a set of array processors. These aggregates are further characterized by the fact that the marking of some members of the aggregate may be relevant to the firing of a successor transition while others may not. This characteristic can model the participation or non-participation of some elements of the array processor in subsequent events; or viewed alternatively, the event resulting in a condition has transpired with the participation and/or non-participation of some elements of the array processor.

Unique to S Nets is the concept of two kinds of firings — one of which — the mask firing — provides for alternatives in the markings of the aggregates. This enables modeling of changes in the participation or non-participation of the elements of an array processor as it proceeds from event to event.

2.2 S NET GRAPHS

S Nets will be defined in terms of sets. The element of a set will be designated within { }. The CARDINALITY of any set shall be designated | | and refers to the number of elements in the set, i.e., $|S|$ refers to the number of elements in S.

If $S = \{s_1, s_2,\ldots s_j\}$, then $|S| = j$.

Also important in the S Net definition is the notion of tuples noted as < > and consisting of ordered components. The cardinality of a tuple is also designated | |, but it is more appropriately called its DIMENSIONALITY.

An S Net Graph will be a quadruple (T,S,U,A), with an initial marking K_0 and a set of transition descriptors D, where:

T = A finite set of transitions $\{t_1, t_2, \ldots t_{|T|}\}$.

S = A finite set of scalar places $\{s_1, s_2, \ldots s_{|S|}\}$.

U = A finite set of vector-mask places $\{<V_1, M_1>, <V_2, M_2>, \ldots <V_{|U|}, M_{|U|}>\}$

A = A finite set of directed arcs $\{a_1, a_2, \ldots a_{|A|}\}$, such that $A \subsetneq (PxT) \cup (TxP)$, where $P = U \cup S$ and P is called the set of places, and the elements of A are of the form $<p_j, t_k>$ or $<t_j, p_k>$, so that either an arc connects a place to a transition or a transition to a place.

Transitions are of two types — those accomplishing a simple firing, and those accomplishing a mask firing. The descriptor for a transition t_i is D $[t_i]$ and designates the transition type. This will be further defined under "Rules for Execution".

The set U is defined as a subset of $\mathcal{V} x \mathcal{M}$, where:

$\mathcal{V}$ = A finite set of elements called vector places $\{V_1, V_2, \ldots V_{|\mathcal{V}|}\}$; each element of $\mathcal{V}$, designated V_i, is a tuple of some number of ordered components, where $V_i = <v_{i1}, v_{i2}, \ldots v_{ip}>$, and $p > 1$, and $|V_i|$ does not necessarily equal $|V_j|$ when $i \neq j$, but $|\mathcal{V}| = |U|$.

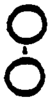
$\mathcal{M}$ = A finite set of elements called masks $\{M_1, M_2, \ldots M_{|\mathcal{M}|}\}$; each element of $\mathcal{M}$, designated M_i, is a tuple of some number of ordered components, where $M_i = <m_{i1}, m_{i2}, \ldots m_{iq}>$, and $q > 1$, and $|M_i|$ does not necessarily equal $|M_j|$ when $i \neq j$, but $|\mathcal{M}| = |U|$.

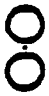
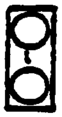
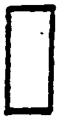
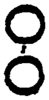
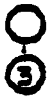
The relation to determine U, a subset of $\mathcal{V}x\mathcal{M}$, is:

$U = \{(V_i, M_i) | V_i \in \mathcal{V}, M_i \in \mathcal{M}$, and $Pr(V_i, M_i)$ is true$\}$

where $Pr(V_i, M_i)$ = "The ith element of $\mathcal{V}$ is associated with the ith element of $\mathcal{M}$, and the jth component of V_i is associated with the jth component of M_i." Since $|\mathcal{V}| = |U|$ and $|\mathcal{M}| = |U|$, then $|\mathcal{V}| = |\mathcal{M}|$, but for every V_i and M_i, $|V_i| \le |M_i|$. (This difference in dimensionality will not alter S Net execution.)

If P is $U \cup S$, and A is as defined, the triple (P,T,A) is a bipartite directed graph since all nodes can be partitioned into two sets, transitions and places, such that each arc directed FROM an element of one set is directed TO an element of the other set, and vice versa. Therefore arcs from(to) a vector-mask place or a scalar place are always directed to(from) a transition.

In the S Net Graph, transitions are represented by ├ , the scalar places are represented by ○ , and the vector-mask places by  .

Within that last symbol, the vector symbol V_i is  and the mask symbol M_i is  or  . The dimensionality of V_i or $|V_i|$ in  is portrayed as  .

The dimensionality of M_i is not noted on the graph, but is specified in the formal designation of M_i components, i.e., $<m_{i1}, m_{i2}, \ldots m_{i|M_i|}>$. Arcs are denoted as ⟶ .

Analogously to a Petri Net, the structure of an S Net is defined so as to make clear the relationship of places and transitions. For each transition $t \varepsilon T$ of the Net, there are two functions defined: the input function I(t), designating the set of input places for the transition; and the output function O(t), designating the set of output places for the transition.

$$I: \ T \longrightarrow 2^{P}, \text{ where } I(t) = \{p \mid <p,t> \varepsilon A\}$$

$$O: \ T \longrightarrow 2^{P}, \text{ where } O(t) = \{p \mid <t,p> \varepsilon A\}$$

The INPUT PLACES of a transition are all the scalar places and vector-mask places directed immediately TO the transition. The OUTPUT PLACES of a transition are all the scalar places and vector-mask places directed immediately FROM the transition.

2.3.1 S Net Example

Given the S Net shown in Figure 3, we shall delineate the graph of the S Net as follows:

$T = \{t_1, t_2, t_3, t_4, t_5, t_6\}$

$S = \{s_1, s_2, s_3\}$

$V = \{V_1, V_2, V_3, V_4\}$

$V_1 = <v_{11}, v_{12}, v_{13}>$

$V_2 = <v_{21}, v_{22}, v_{23}>$

$V_3 = <v_{31}, v_{32}, v_{33}>$

$V_4 = <v_{41}, v_{42}, v_{43}>$

$M = \{M_1, M_2, M_3, M_4\}$

$M_1 = <m_{11}, m_{12}, m_{13}>$

$M_2 = <m_{21}, m_{22}, m_{23}>$

$M_3 = <m_{31}, m_{32}, m_{33}>$

$M_4 = <m_{41}, m_{42}, m_{43}>$

$U = \{<V_1, M_1>, <V_2, M_2>, <V_3, M_3>, <V_4, M_4>\}$

$A = \{<s_1, t_1>, <t_1, <V_1, M_1>>, <<V_1, M_1>, t_2>, <t_2, <V_2, M_2>>,$
$<<V_2, M_2>, t_3>, <t_3, s_2>, <t_1, <V_3, M_3>>, <<V_3, M_3>, t_4>, <t_4, <V_4, M_4>>,$
$<<V_4, M_4>, t_3>, <s_2, t_5>, <t_5, s_1>, <s_2, t_6>, <t_6, s_3>\}$

2.4 MARKINGS

The infinite set of non-negative integers {0, 1,...} is designated N; the set of Boolean numbers {0, 1} is designated B; the set {0} is designated Z; and the set {1} is designated L. Also the r-fold Cartesian products are defined:

N^r = NxNx...N; each member is of the form $<N_1, N_2, \ldots N_r>$

B^r = BxBx...B; each member is of the form $<B_1, B_2, \ldots B_r>$

Z^r = ZxZx...Z; each member is of the form <0, 0,...0>

L^r = LxLx...L; each member is of the form <1, 1,...1>

where N_i and B_i, are elements of N and B, respectively.

A MARKING is a function K where

$$K: \; S \rightarrow N; \; V_i \rightarrow N^{|V_i|} \text{ for all } V_i \; \varepsilon \; \mathcal{V}; \; M_i \rightarrow B^{|M_i|} \text{ for all } M_i \; \varepsilon \; \mathcal{M}.$$

The marking associates a non-negative integer with each scalar place — K(s) for each $s \; \varepsilon \; S$ — and two vectors of non-negative integers with each vector-mask place, one of those associated vectors being a Boolean vector — $K(V_i)$ for each $V_i \; \varepsilon \; \mathcal{V}$ and $K(M_i)$ for each $M_i \; \varepsilon \; \mathcal{M}$.

A marking for an S Net must specify all three components. Henceforth, when the function K is specified and the mapping is for S, then K_S is being given; when the mapping is for $\mathcal{V}$, then $K_{\mathcal{V}}$ is being given; and when the mapping is for $\mathcal{M}$, then $K_{\mathcal{M}}$ is being given.

An INITIAL MARKING K_0 is defined as the first marking of the S Net .

A MASK MARKING for a mask M_i is a function K such that $M_i \rightarrow 3^{|M_i|}$. The set of possible mask markings for any M_i is $W(M_i)$ and denotes the co-domain of a mask marking, consisting of designated tuples of $3^{|M_i|}$, or if appropriate, the entire product set $3^{|M_i|}$.

2.4.1 Notation for Markings

The convention adopted to show markings will be that of ( ) and < >. The former is used to distinguish the marking of a single element or component, and the latter is used when more than one element or component is involved, thereby denoting an ordering with respect to markings of elements or components.

As an example, markings for places s_1, s_2, s_3 are designated $K(s_1) = (0)$; $K(s_2) = (0)$; $K(s_3) = (0)$, and if $S = \{s_1, s_2, s_3\}$, this is equivalent to $K\colon\ S \rightarrow Z$.

By definition V and M are sets consisting of elements which are tuples. If $V = \{V_1, V_2\}$, and if $V_1 = <v_{11}, v_{12}, v_{13}>$ and $V_2 = <v_{21}, v_{22}>$, then $K: V_1 \rightarrow Z^3; V_2 \rightarrow Z^2$ is equivalent to:

$K(V_1) = <0,0,0>$; alternately $K(v_{11}) = (0)$; $K(v_{12}) = (0)$; $K(v_{13}) = (0)$; and $K(V_2) = <0, 0>$; alternately $K(v_{21}) = (0)$; $K(v_{22}) = (0)$. Similarly, for $M_1 = <m_{11}, m_{12}, m_{13}>$, a marking $K(M_1) = <1, 0, 0>$ designates that $K(m_{11}) = (1)$, $K(m_{12}) = (0)$, $K(m_{13}) = (0)$.

2.4.2 Graphic Portrayal of Markings

Markings are illustrated with the presence of tokens. Dots in any place represent tokens. Tokens in masks may, alternatively, be represented by Boolean symbols for legibility.

The symbol $\boxed{\begin{matrix}1\\1\\0\end{matrix}}$ for M_1 shows a token in m_{11} and m_{12}. This is synonomous with the symbol $\boxed{\begin{matrix}\odot\\\odot\\\bigcirc\end{matrix}}$.

Given the S Net example of Figure 3, Figure 4 illustrates an ~~initial~~ marking where masks are marked with tokens but vector places are not marked with tokens. This example assumes the initial marking is:

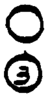
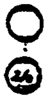
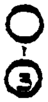
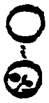
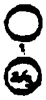
$K_0(s_1) = (1)$; $K_0(s_2) = (0)$; $K_0(s_3) = (0)$

$K_0(V_1) = K_0(V_2) = K_0(V_3) = K_0(V_4) = <0, 0, 0>$

$K_0(M_1) = K_0(M_2) = <1, 0, 0>$

$K_0(M_3) = <1, 1, 1>$

$K_0(M_4) = <0, 0, 1>$

An assignment of tokens to a vector place V_i may leave some of the component places marked with tokens and others empty, i.e., all elements v_{ij} of V_i may not have tokens. Since the $|V_i|$ may be large, graphic designation of which components are marked must necessarily be limited. For example, a  portrays a V_i with three components, then  indicates that two $v_{ij} \varepsilon V_i$ are marked. Syncnomous are the symbols  and  . However  only conveys that two v_{ij} are marked but does not distinguish the individual elements, nor does it indicate how many tokens are in each v_{ij} marked. However S Nets use vector-mask places to model conditions resulting from and leading to events, and V_i is always expressed graphically in conjunction with M_i. It is M_i which will be used graphically to enhance comprehension of which v_{ij} are marked — at least in markings resulting from an execution of the S Net.

2.5 RULES FOR EXECUTION

The graph and structure of S Nets have been addressed in previous Sections of this Chapter. Now discussed will be the dynamic behavior of S Nets.

2.5.1 Enabled Transitions

A scalar place is HOLDING if it has at least one token in it.

A vector-mask place $<V_i, M_i>$ is HOLDING if:

at least one $K(m_{ij}) = (1), j=1,2,...|V_i|$, and

$v_{ij} \varepsilon V_i$ has a non-zero marking for all those j for which $m_{ij} \varepsilon M_i$ has a non-zero marking.

A holding for a vector-mask place is in contrast to a marking of that place. Whereas a marking associates some set of integers with vector-mask places, a holding for a vector-mask place REQUIRES that the components of V_i be marked with tokens everywhere that their associated M_i components are marked with tokens.

A transition t is ENABLED also called FIRABLE under the following conditions: t is ENABLED if all scalar places in I(t) are holding and all vector-mask places in I(t) are holding.

2.5.2 Firing Transitions

A FIRING is a function of a transition which has for its domain and range the marking of the input and output places of the transition. There is a firing associated with every enabled transition t. When a transition t is enabled, its firing function is defined at a given marking K_n of the S Net, and the firing yields K_{n+1}, a new marking.

2.5.3 Transition Types

A TRANSITION TYPE specifies the firing capabilities of the transition — either simple or mask firing — designated SFT and MFT respectively.

2.5.4 Transition Descriptors

A TRANSITION DESCRIPTOR D[t] specifies the transition type, either SFT or MFT, and for every vector-mask output place $<V_i, M_i>$ of the transition, specifies $W(M_i)$, the set of markings for M_i. Descriptors for a transition t with vector-mask output places $<V_i, M_i>, <V_j, M_j>, \ldots <V_r, M_r>$ are specified:

$$D[t] = [\text{type}; K(M_i) \epsilon W(M_i), K(M_j) \epsilon W(M_j), \ldots K(M_r) \epsilon W(M_r)]$$

where the elements of $W(M_i)$ are explicitly listed, as:

$$D[t] = [\text{type}; K(M_i) \epsilon \{< >, < >, \ldots\}, K(M_j) \epsilon \{< >, < >, \ldots\}, \ldots K(M_r) \epsilon \{< >, < >, \ldots\}]$$

2.5.5 Rules for a Simple Firing

A SIMPLE FIRING associated with an enabled transition t is such that:

For every scalar input place s, then

$K_{n+1}(s) = K_n(s) - 1$

For every scalar output place s, then

$K_{n+1}(s) = K_n(s) + 1$

For every vector-mask input place $<V_i, M_i>$, then: for $v_{ij} \varepsilon V_i$, $j=1,2,\ldots|V_i|$,

$K_{n+1}(v_{ij}) = K_n(v_{ij}) - 1$ for those j for which $m_{ij} \varepsilon M_i$ has a non-zero marking; and for $m_{ij} \varepsilon M_i$,

$K_{n+1}(m_{ij}) = K_n(m_{ij})$ for all j.

For every vector-mask output place $<V_i, M_i>$, then: for $v_{ij} \varepsilon V_i$, $j=1,2,\ldots|V_i|$,

$K_{n+1}(v_{ij}) = K_n(v_{ij}) + 1$ for those j for which $m_{ij} \varepsilon M_i$ has a non-zero marking; and for $m_{ij} \varepsilon M_i$,

$K_{n+1}(m_{ij}) = K_n(m_{ij})$ for all j.

As seen from the firing rules, SFTs do not alter their input or output masks.

2.5.6 Rules for a Mask Firing

A MASK FIRING is associated with an enabled transition t that has at least one $<V_i, M_i>$ output place, and is such that:

For every scalar input place s, then

$K_{n+1}(s) = K_n(s) - 1$

For every scalar output place s, then

$K_{n+1}(s) = K_n(s) + 1$

$j=1,2,\ldots|V_i|$,

$K_{n+1}(v_{ij}) = K_n(v_{ij}) - 1$ for those j for which $m_{ij} \varepsilon M_i$ has a non-zero marking; and for $m_{ij} \varepsilon M_i$,

$K_{n+1}(m_{ij}) = K_n(m_{ij})$ for all j.

For every vector-mask output place $<V_i, M_i>$, then for M_i

$K_{n+1}(M_i) \varepsilon W(M_i)$, where $W(M_i)$ is specified by the transition descriptor $D[t]$, and $K_{n+1}(M_i)$ is non-deterministically chosen.

For every vector-mask output place $<V_i, M_i>$, then for $v_{ij} \varepsilon V_i$,

$j=1,2,\ldots|V_i|$,

$K_{n+1}(v_{ij}) = K_n(v_{ij}) + 1$ for those j for which $m_{ij} \varepsilon M_i$ has a non-zero marking, i.e., where $K_{n+1}(m_{ij}) = (1)$.

The assignment of a Boolean vector to M_i by MFT t is a mapping of M_i INTO $W(M_i)$, where the domain is M_i and the co-domain consists of the elements of $W(M_i)$.

$$M_i \xrightarrow{\text{INTO}} W(M_i)$$

$$M_j \xrightarrow{\text{INTO}} W(M_j)$$

By the firing definitions, firings remove tokens from places and add tokens to other places, and in the case of the mask firing mark the masks of the vector-mask output places. It should be noted that the number of tokens subtracted by a transition firing does not necessarily equal the number that it adds.

2.5.7 Transitions and Their Descriptors

As seen from the firing definitions, SFTs on firing do not change the $K(M_i)$ of their $<V_i, M_i>$ input and output places. Where the firing type is simple, the markings for any output masks of the transition are specified at K_0. This transition descriptor is noted simply as $D[t] = [SFT; \_]$.

For MFTs, the $|W(M_i)| \geq 1$ for all output masks, and since these markings are accomplished by the transition firing and not the initial marking, the set of markings must be listed in the transition descriptor, i.e.,

$D[t] = [MFT; K(M_i) \varepsilon \{< \quad >, < \quad > \quad \}]$

2.5.8 Graphic Portrayal of Mask Firing Transitions

Figure 5 illustrates a MFT t_1 with one output mask M_1 and two designated markings of that mask, i.e.,

$D[t_1] = [MFT; K(M_1) \varepsilon \{<1, 0, 0>, <1, 1, 0>\}]$.

Figure 6 shows a mask firing of two output masks M_1 and M_2 with two possible markings for each mask, i.e.,

$D[t_1] = [MFT; K(M_1) \varepsilon \{<1, 0, 0>, <1, 1, 0>\}, K(M_2) \varepsilon \{<1, 1>, <0, 1>\}]$.

At the time of firing, the mask marking that is accomplished by the transition is arbitrary. According to the firing rules stated for the MFT, M_i is marked with one tuple of the set $W(M_i)$. In Figure 5 after firing of t_1, the $D[t_1]$ allows markings $K(M_1) = <1, 0, 0>$ or $K(M_1) = <1, 1, 0>$.

2.5.9 Example of Mask Firing Transitions

Figure 7 through Figure 10 show MFTs in an S Net and illustrate how they perform. Given an initial marking:

$K_0(s_1) = (1)$: $K_0(s_2) = (0)$; $K_0(s_3) = (0)$

$K_0(V_1) = K_0(V_2) = <0, 0, 0>$

and Descriptors:

$D[t_1] = [MFT; K(M_1) \epsilon \{<1, 0, 0>, <0, 0, 1>\}]$

$D[t_2] = [MFT; K(M_2) \epsilon \{<1, 0, 0>, <0, 0, 1>\}]$

$D[t_3] = D[t_4] = D[t_5] = [SFT;\_]$

Figure 7 reflects the initial marking and shows that t_1 is enabled. Transition t_1 then fires, and the results are shown in Figure 8 . The marking assigned for M_1 by t_1 was <1, 0, 0> which is designated in $D[t_1]$ and is shown on the graph as one member of the set $W(M_1)$. After t_1 fires, v_{11} receives one token since m_{11} is marked with a token, and a token is removed from s_1.

The v_{11} token enables t_2 since m_{11} also holds a token, so that the firing of t_2 can commence. If t_2 fires changing the marking $K(M_2)$ to <1, 0, 0>, and if the firing sequence t_1, t_2, t_3, t_4 is assumed, then Figure 9 illustrates the marking after t_4 has fired. In Figure 9 a token is again in s_1; $K(M_1)$ is <1, 0, 0> from the previous firing of t_1; $K(M_2)$ is <1, 0, 0> as marked from the previous firing of t_2; $K(V_1)$ is <0, 0, 0> since the token in v_{11} placed there as a result of the first t_1 firing was removed at the firing of t_2. $K(V_2)$ is <0, 0, 0> since the token in v_{21} placed there after the firing of t_2, was removed at the firing of t_3.

Where Figure 9 shows t_1 enabled, Figure 10 shows the results after the second firing of t_1. Here $K(M_1) = <0, 0, 1>$, a marking alternative also described in $D[t_1]$. (The selection of the marking is arbitrary, and could have been <1, 0, 0> again.) Given $K(M_1) =$ <0, 0, 1>, by firing definition v_{13} now receives a token. Since m_{13} also holds a token, t_2 is enabled, and so on.

3. S NET APPLICATION

Before proceeding with an illustrative application one more definition is provided. An INTERPRETATION of an S Net is an assignment of labels to the transitions and/or places of the Net to indicate for the transition the event that it models and for the place the condition that it models.

Consider the problem of summing the rows of a 4x4 matrix A, multiplying the resultant sum by a 4xl vector B, then storing the results in the first column of A. This is described with the FORTRAN algorithm:

```
          DO 200 I = 1, 4
          DO 100 J = 1, 3
      100 A(I, 1) = A(I, 1) + A(I, J+1)
      200 A(I, 1) = A(I, 1) * B(I)
```

If it is assumed that an 8 PE SIMD machine is available for this computation, some data storage scheme for the vectors needs to be developed. Figure 11 depicts one such scheme. The Figure indicates that PE_0 has the first row in its PEM, PE_1 has the second row, etc. With parallel hardware available, the four row sums can be formed simultaneously, and then all four sums multiplied by the vector also simultaneously. This is shown in an S Net model in Figure 12 which is marked to reflect a holding of condition after a t_2 firing.

Transition t_2 models the event which adds the Jth column of matrix A to the first column of matrix A. Assuming a sequence of transition firings such that t_4 and t_5 have conflict resolved by the status of loop counter J, when t_4 finally fires, all rows are summed; and transition t_7 models a parallel multiplication of all four sums by the appropriate element of vector B. Something less than the maximum utilization of the vector resource is indicated when the S Net is put into this sequence of firings. The marking on the vector-mask output place resulting from the t_2 firing is depicted on Figure 12 as 4/8. Thus the utilization of the array resources is a by-product of an S Net execution. Both the parallelism achieved by the array resource (4) and the utilization with respect to the maximum parallelism supportable by the array hardware (8) becomes apparent. Finally, the marking of the mask suggests some additional management activity required of the vector resource. It is necessary to specify which of the PEs available will participate in the process and which will not.

The S Net mask models control over the array resources, the designation of which PEs are participating in events and which are not. The event modeled by t_2 is an array operation since the output place of t_2 is a vector-mask output place $<V_1, M_1>$. This event takes place with the first four PEs of the array resource participating and the latter four non-participating since $K(M_1) = <1^4, 0^4>$. The add operation (the event modeled by t_2) is performed by the four array resources simultaneously and results in a condition viewed as a logical aggregate -- a vector condition, i.e., the Jth column of matrix A has been added to the first column of matrix A for all PEs participating.

If t_2 had a scalar output place (it does not) as well as a vector-mask output place, it would model an event performed by a scalar resource concurrent with an event performed by an array resource.

4. SUMMARY

In this paper, a new modeling system -- that of Synchronous Nets or S Nets -- has been developed as a modification of Petri Nets. S Nets are analogous to Petri Nets in that they model occurrences of events with the firings of transitions, and model the holding of conditions with the presence of tokens in places. However, S Nets are a modeling tool particularized to express algorithms for SIMD and MSIMD architectures, providing richer detail for the SIMD environment than do Petri Nets. S Nets have introduced vector-mask places to model the conditions of a set of array processors. These places are distinguished from scalar places which model scalar resources. The array resources are modeled as logically associated and homogeneous aggregrates by a single vector-mask place. The underlying parallelism of the array hardware is modeled by the dimensionality of the vector element of the place, and the mask is used to model control of the aggregate as to the designation of which members of the aggregate will participate in the firing of subsequent events, modeled by transitions.

More exposition of the modeling capability of S Nets, particularly the properties of concurrency and conflict, as well as the relationship of S Nets to Petri Nets is supplied in [9], as are additional examples.

The richer detail of S Nets enables the differentiation of Flynn's classes of architectures. Because of the increased modeling power lent by S Nets with vector-mask places, the multiplicity of the data stream can be depicted. Figure 13 illustrates the S Net modeling of the architectural classes.

The SIMD architecture is readily distinguished from the SISD just by the added detail of vector-mask places. The MIMD-2 architecture which allows both scalar and vector concurrency, such as that of the BSP [10], is distinguishable from the MIMD-3 architecture which allows concurrent SIMD resources, such as that of Siegel's PASM [11]. Both are clearly different from the MIMD-1 architecture of the conventional distributed processors.

BIBLIOGRAPHY

1. Flynn, Michael J., "Very High-Speed Computing Systems", Proceedings of the IEEE, Volume 54, No. 12, December 1966, pp. 1901-1909.

2. Thurber, Dennis J., and Wald, L. D., "Associative and Parallel Processors", Computing Surveys, Volume 7, No. 4, December 1975, pp. 215-255.

3. Ruben, Sherwin, et al., "Application of a Parallel Processing Computer in LACIE", Proceedings of the 1976 International Conference on Parallel Processing, pp. 24-32.

4. Krygiel, Annette J., "An Implementation of the HADAMARD Transform on the STARAN Associative Array Processor", Proceedings of the 1976 International Conference on Parallel Processing, p. 34.

5. Daley, J. S., and Underwood., B. D., "Short-Term Weather Prediction on ILLIAC IV", Proceedings of the 1975 Sagamore Computer Conference on Parallel Processing, p. 240.

6. Petri, Carl A., "Kommunikation mit Automaten". Translation by C. F. Greene, Supplement 1 to RADC-TR-65-337, Volume 1, Rome Air Development Center, Griffiss AFB, New York, 1962.

7. Holt, A. W., et al, "Information System Theory Project", Applied Data Research, Inc., RADC-TR-68-305, Rome Air Development Center, Griffiss AFB, New York, September 1968.

8. Holt, A. W., and Commoner, Frederic, "Events and Conditions", Applied Data Research, Inc., New York, 1979, also Record of Project MAC, Conference on Concurrent Systems and Parallel Computation, ACM, New York, 1970.

9. Krygiel, Annette J., "Synchronous Nets for Single Instruction Stream - Multiple Data Stream Computers", D. Sc Dissertation, Sever Institute of Technology, Washington University, St. Louis, MO, May 1980.

10. Stokes, Richard A., "Burrough's Scientific Processor", High Speed Computer and Algorithm Organization, Academic Press, 1977, pp. 85-90.

11. Siegel, H. J., et al, "An SIMD/MIMD Multimicroprocessing System for Image Processing and Pattern Recognition:, Proceedings of the IEE Conference on Pattern Recognition and Image Processing, August 1979, pp. 214-224.

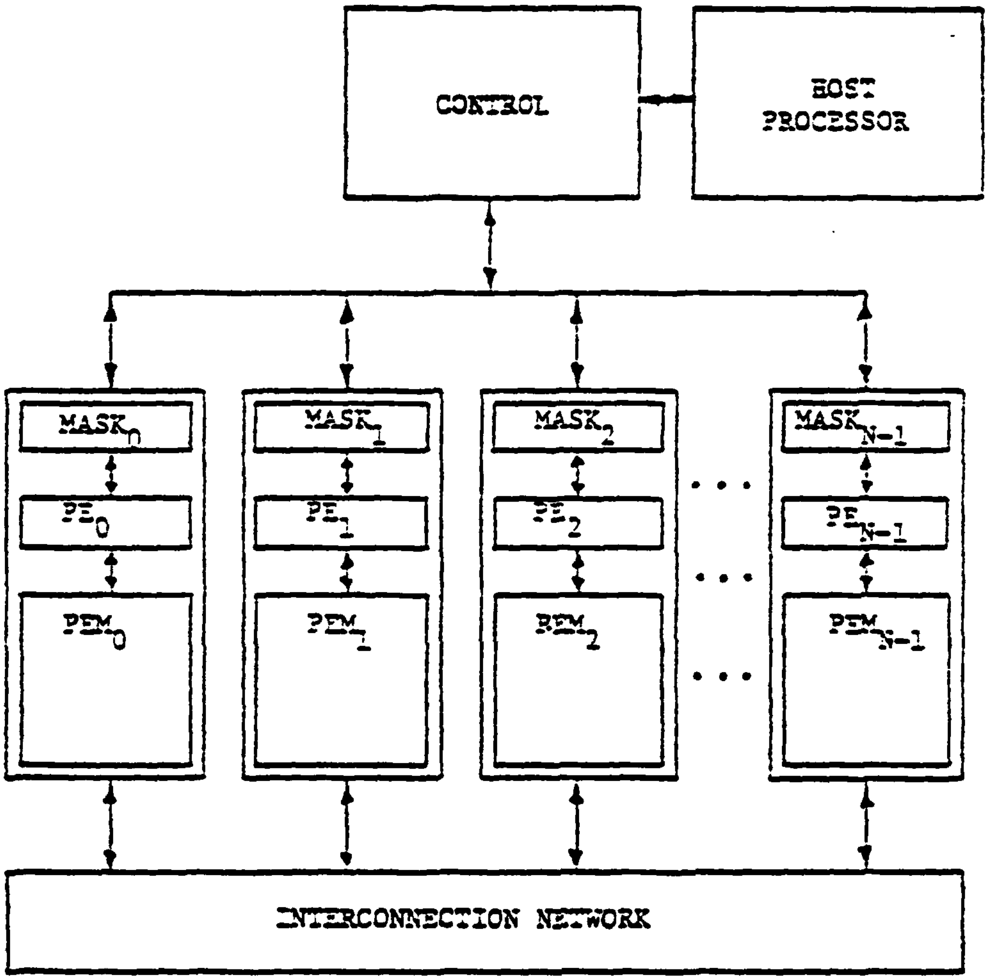


Figure 1 SIMD Architecture

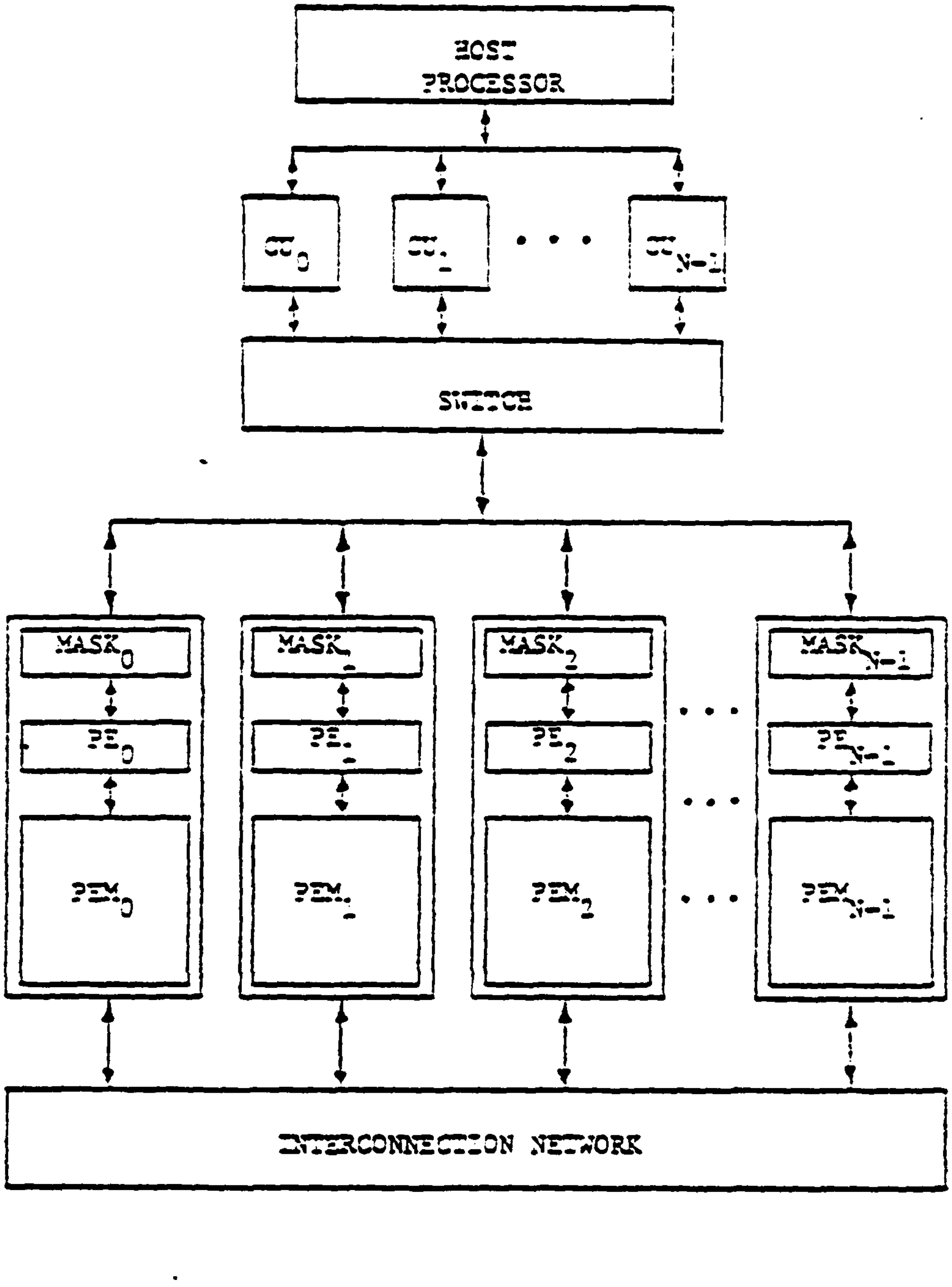


Figure 2 MSIMD Architecture

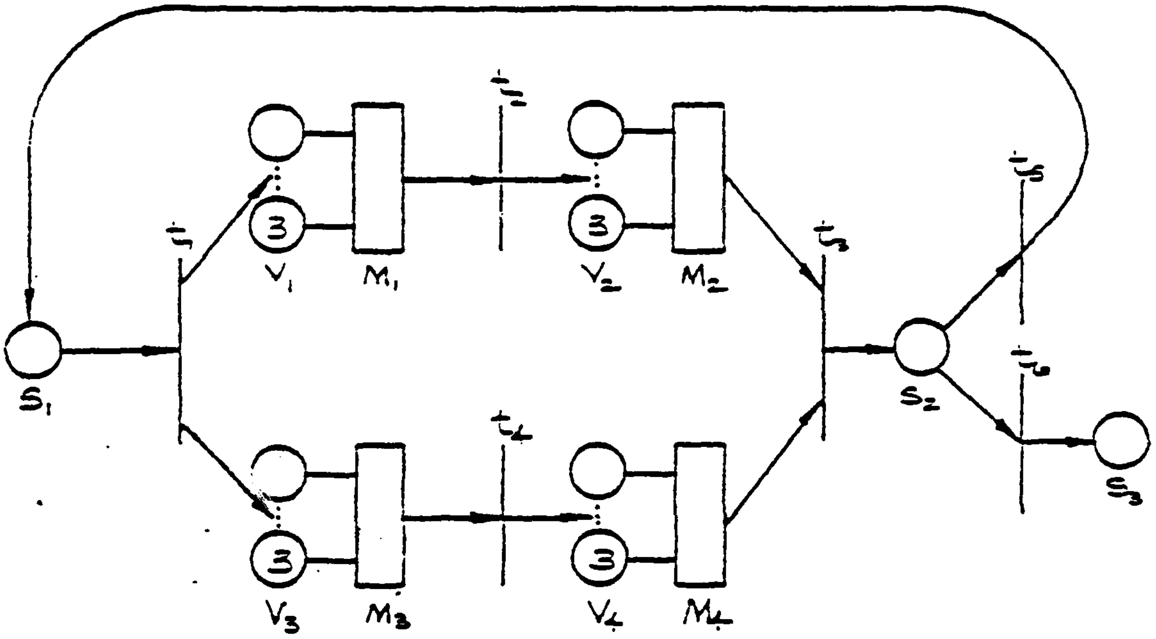


Figure 3 An S Net

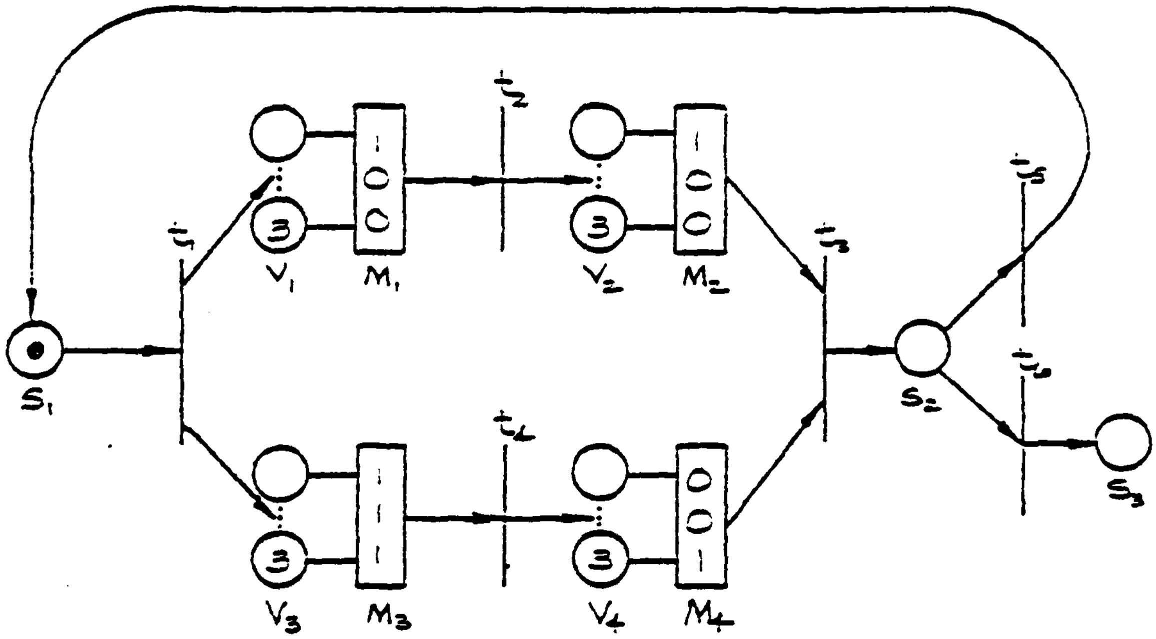


Figure 4 An S Net With M_1 Marked

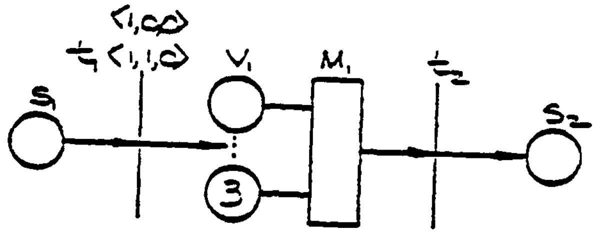


Figure 3 MFT Σ_1 With One Output Mask

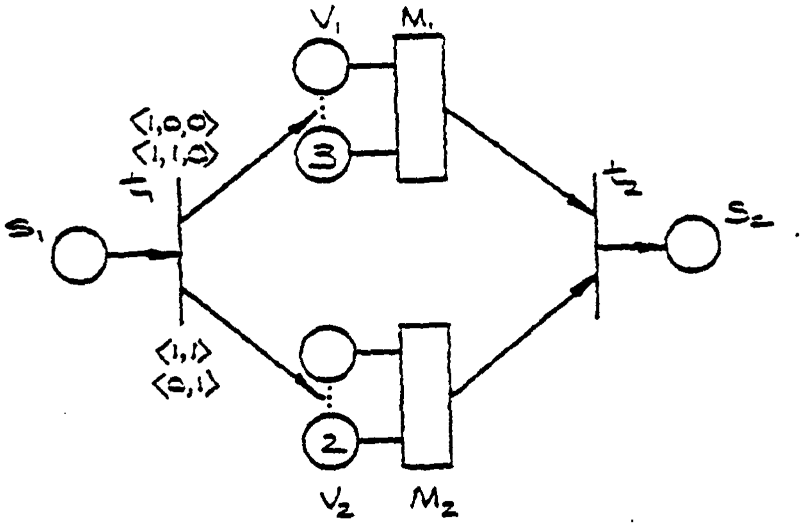


Figure 6 — MFT t_1 With Two Output Masks

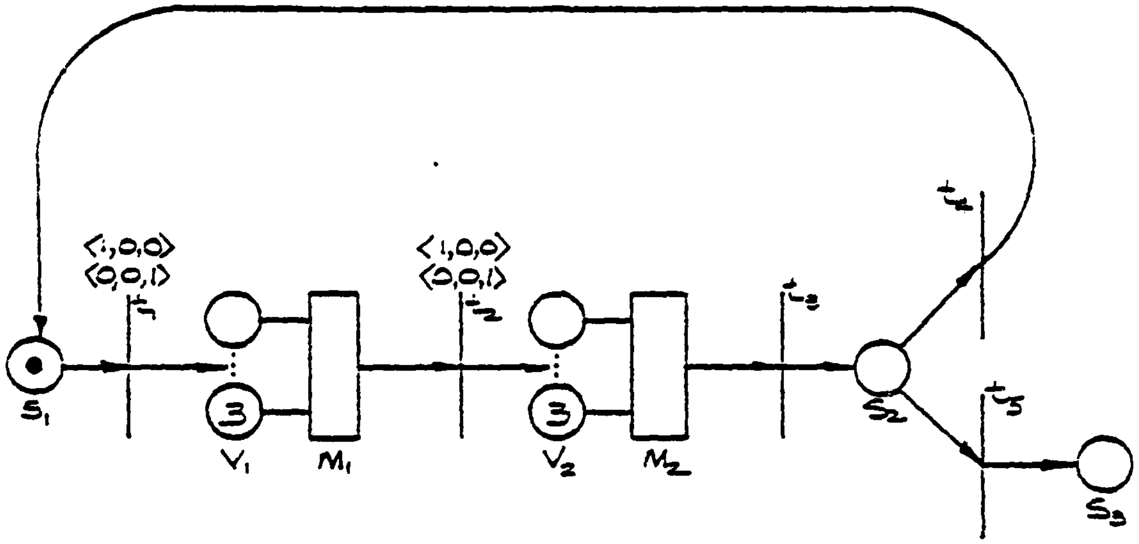


Figure 7 MFTs t_1 and t_2 at K_0

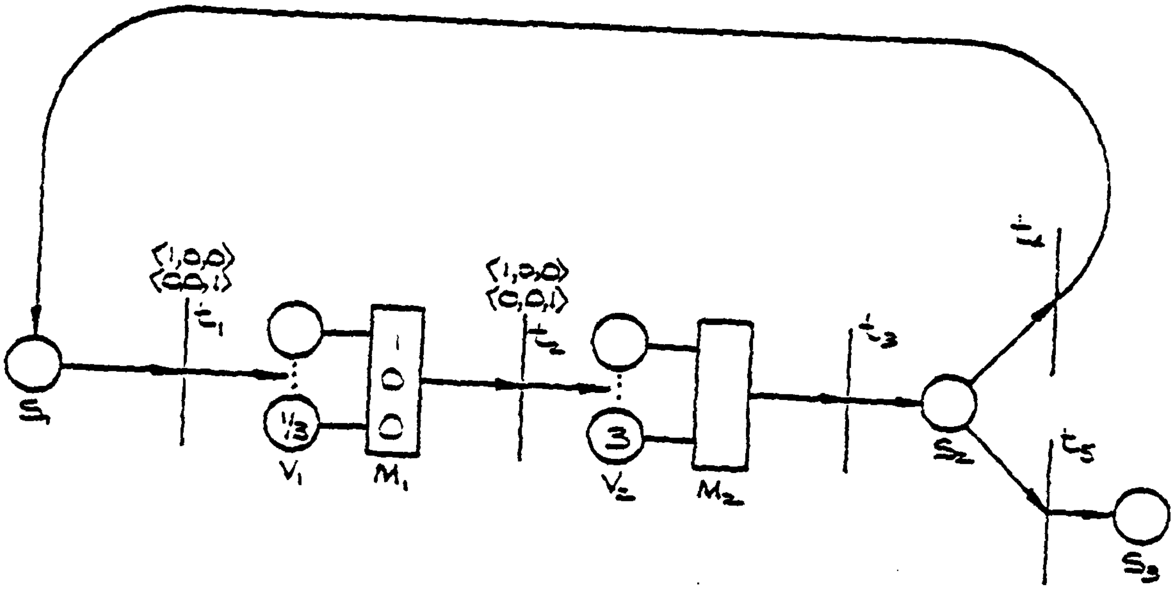


Figure 8 MFTs t_1 and t_2 at K_1

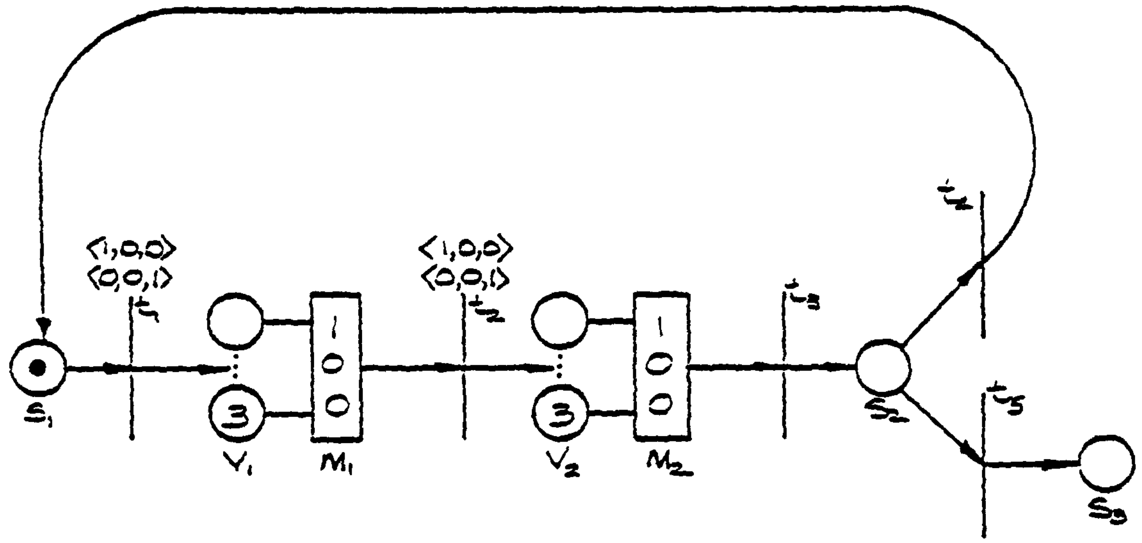


Figure 9 MFTs t_1 and t_2 at K_4

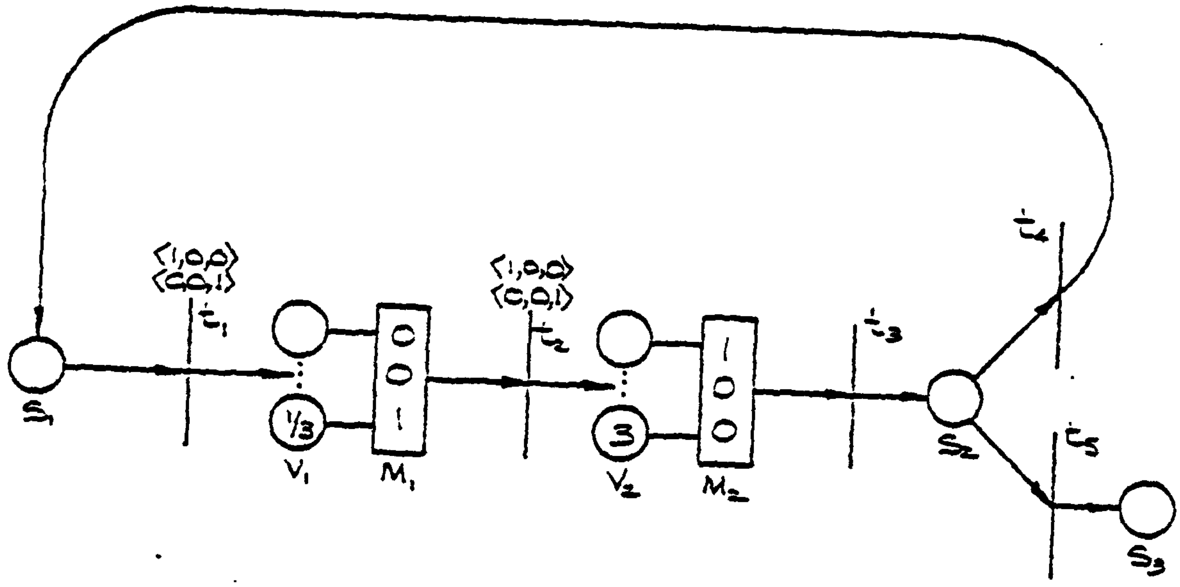


Figure 10 MFTs t_1 and t_2 at K_5

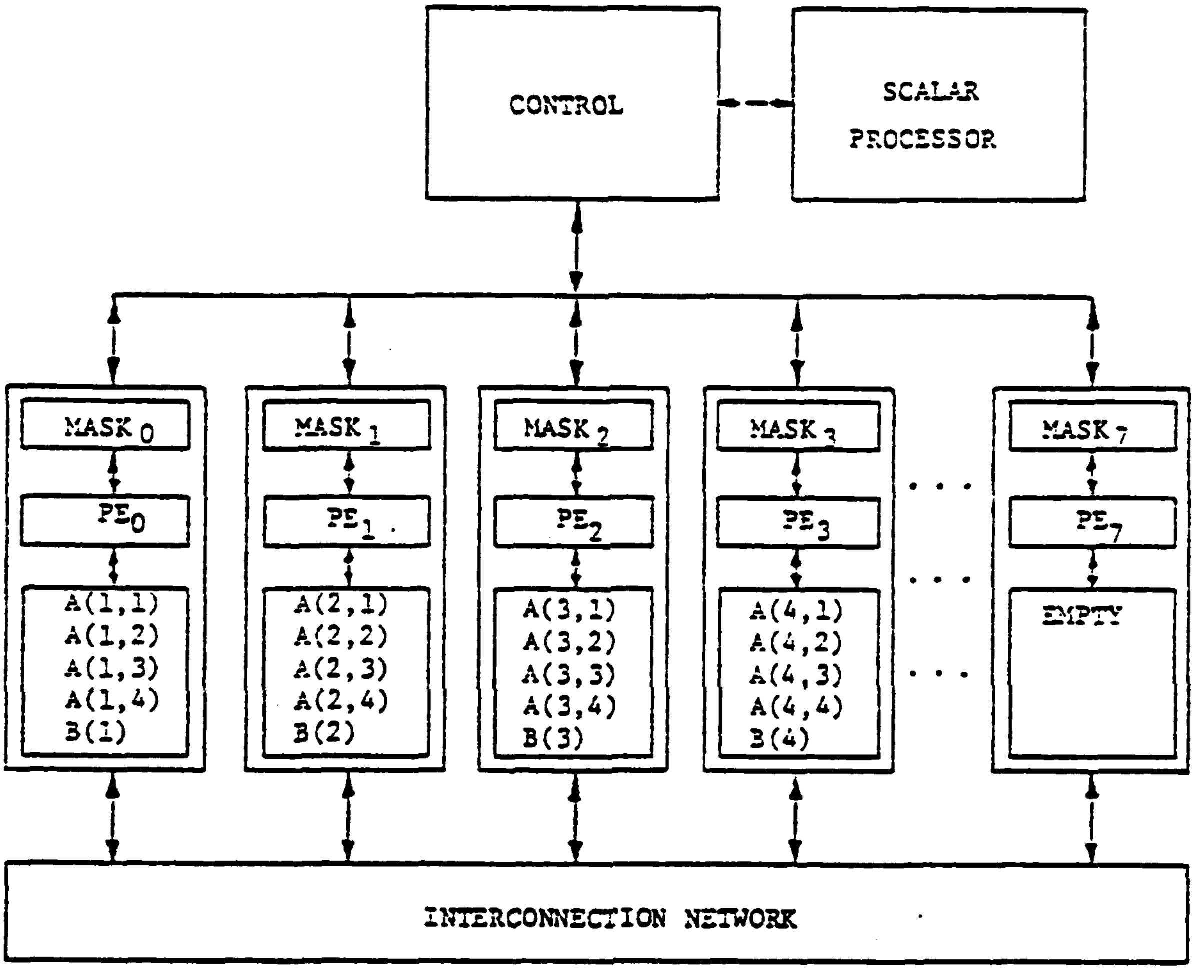


Figure 11 Data Storage Scheme for Row Sum

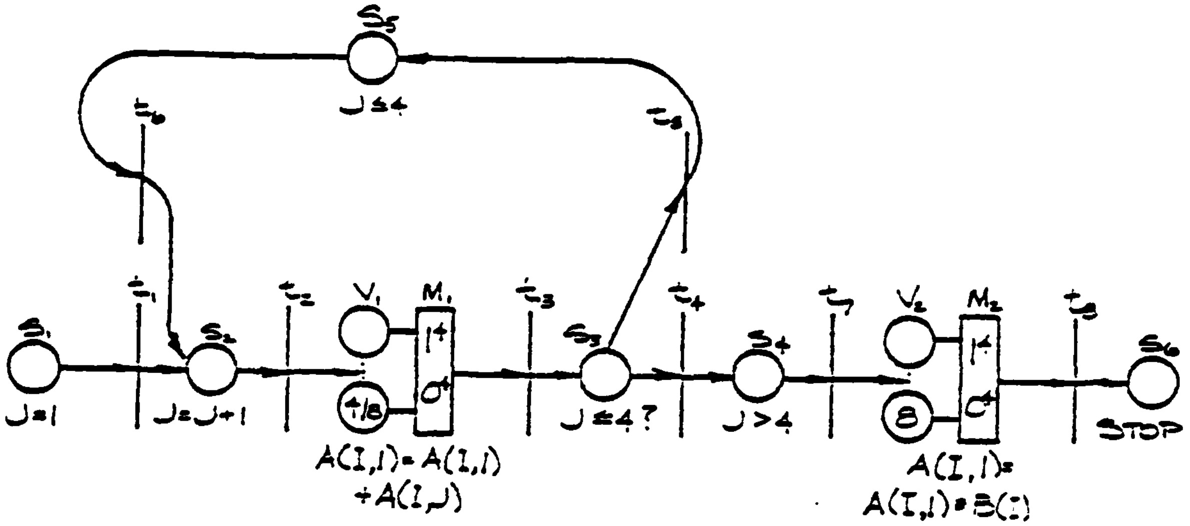
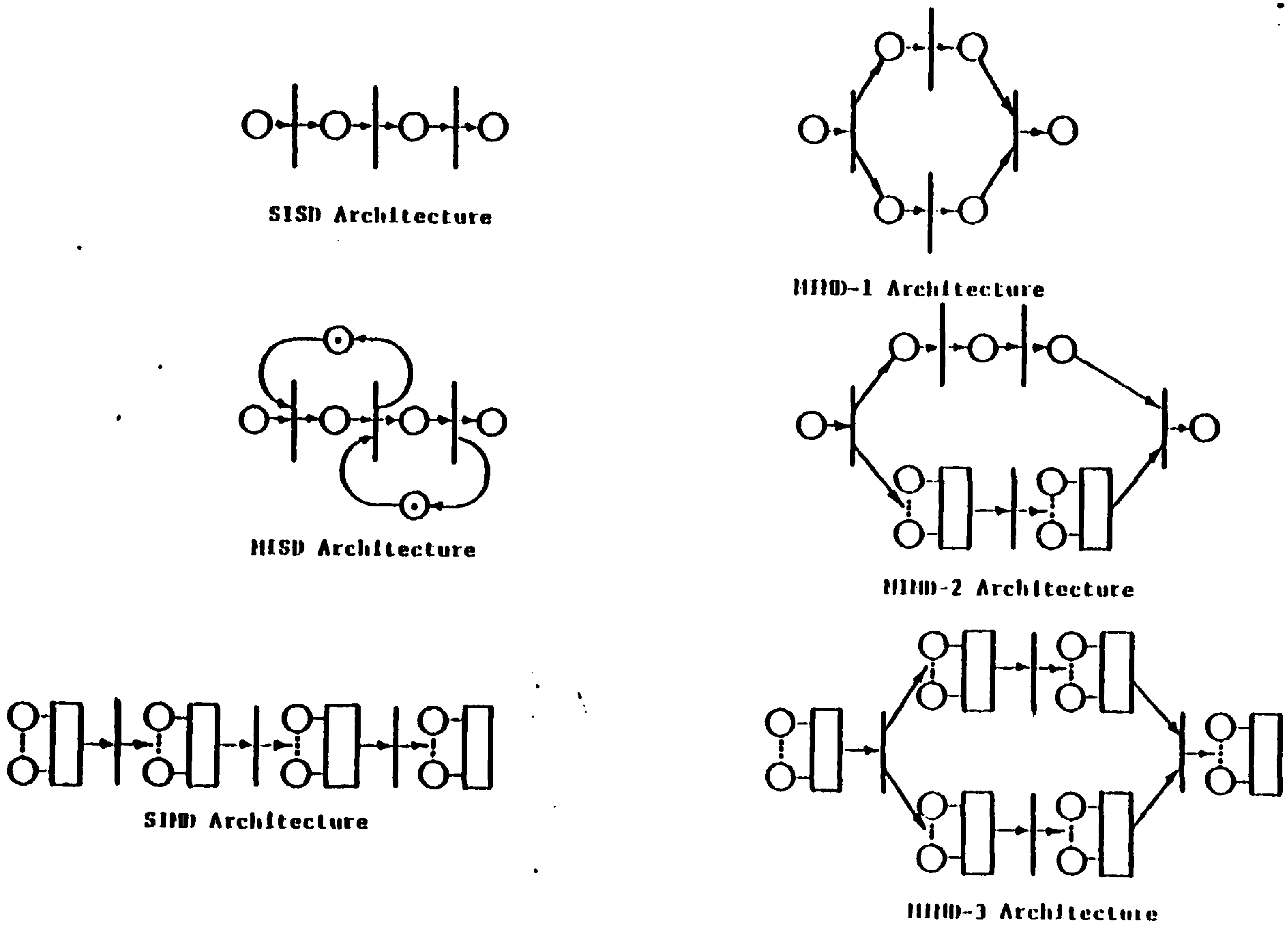


Figure 12 S Net Model of Row Sum



SISD Architecture

MISD Architecture

SIMD Architecture

MIMD-1 Architecture

MIMD-2 Architecture

MIMD-3 Architecture

Figure 13 S Nets Depicting Machine Architectures

DATE

ILME